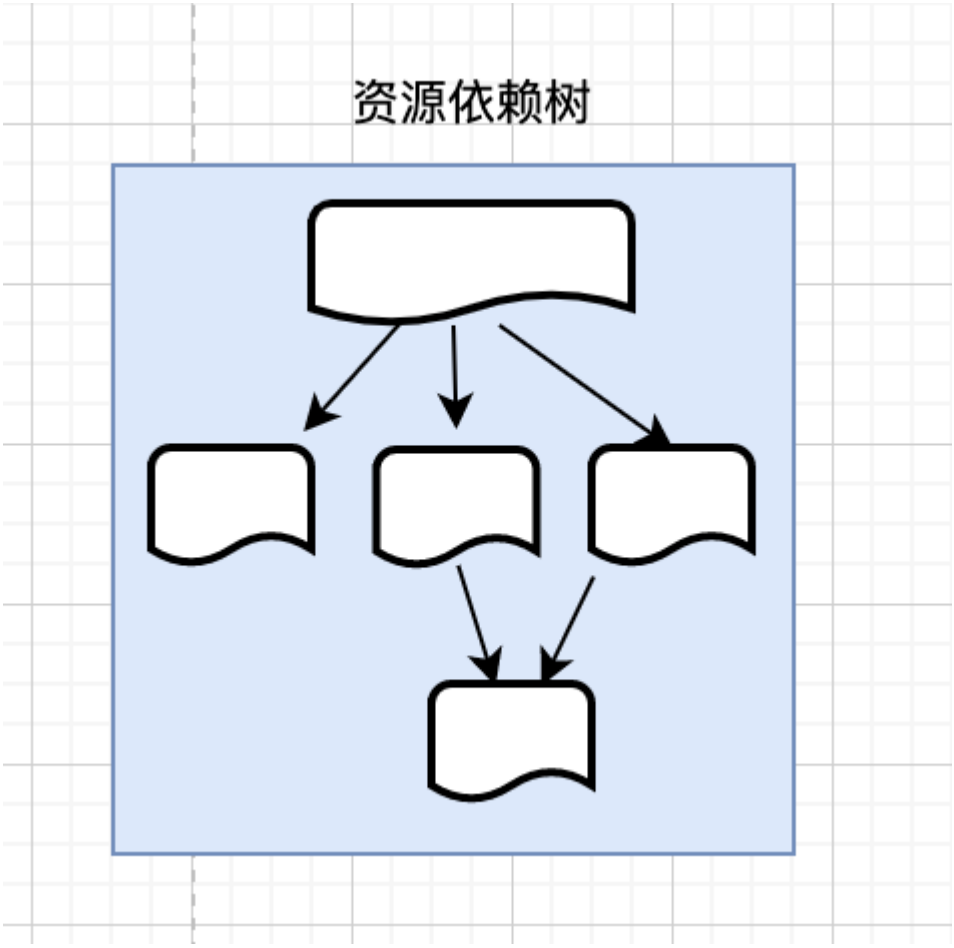
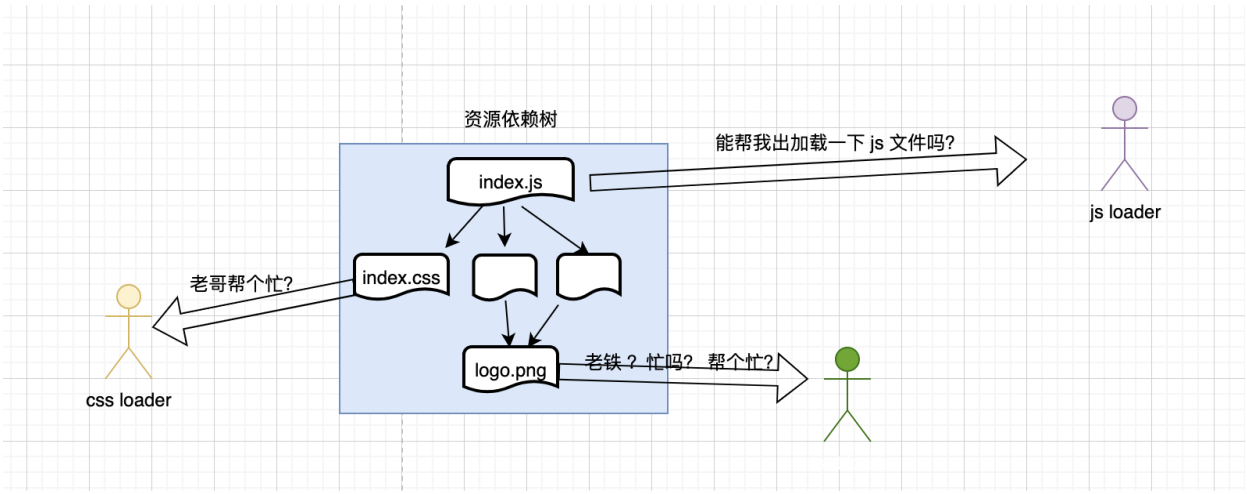
webpack是想要实现整个前端项目的模块化，那各种各样的资源都应该被当成模块来处理。

webpack的加载模式的简单工作流

1. webpack为了更好记录各个文件的依赖关系，webpack内部会生成一颗树状数据结构来记录，这与文件系统的树状结构十分类似，这颗树的里还未载入”模块的内容”，可以理解为暂时都是空的节点。



2. 空的依赖树很无趣的，因为没有办法做任何事情，这个时候，我们需要向这颗空的资源树内填入内容（遍历整个树，赋值的操作）。
3. 遍历赋值的工程中，当然不是像刷leetcode一样简单赋值。而是需要请求loader得帮助，loader会将加载过的资源，返回给节点，节点会存储自己的对象当中。



4. 经过赋值之后的树，就变得十分有内涵了，就彻底成为一个合格的“资源树”。树中的每个节点，都有加载进来的文件内容。

一些思考

1. 设计模式：webpack此处的设计模式跟“模版模式”十分的相似。整个一套流程下来是固定的，只不过在加载特定“资源”的时候，需要做特定的处理。“Loader.js” 更像是实现了模版模式中的某个接口，使得webpack在固定的运行逻辑基础之上，有了更强的拓展性，使用者可以根据不同的场景开发出合适的加载器。
2. Loader的串行使用是有可能的，当Loader都面向特定接口去实现的时候，loader的输入和输出都是明确的，这使得Loader的串行使用可以落地实现。即 一个loader的输出，可以成为另外一个loader的输入，使得拓展性功能更加丰富，多个loader可以实现相互的配合来实现更加牛逼的加载效果。